COPYRIGHT PROTECTION OF COMPUTER

SADULLA KARJIKER†
Associate Professor in Mercantile Law, Stellenbosch University

This article seeks to address a misconception concerning the scope of copyright protection of computer programs. It has been suggested that unlike the US courts, the UK and South African courts have not drawn a proper distinction between functional works — such as computer programs — and other copyright works, with the result that they are more likely to protect ideas, rather than their particular expression. While this may have been true at some stage, it is certainly not the current position in the UK. The decision of the court in Navitaire Inc v easyJet Airline Company & another, *and subsequent decisions, represented a sea-change in UK copyright law relating to computer programs, resulting in comparatively thin copyright protection for computer programs, which corresponds to the legal position in the US. It was the recognition of the functional nature of computer programs that led to this change in how programs should be assessed in terms of copyright doctrine. This more limited protection is considered to strike an appropriate balance between providing the necessary incentives for the production of computer programs, while allowing for a sufficiently large public domain.*

I INTRODUCTION

Copyright protection has been extended to computer programs in South Africa since as far back as 1981.[1] However, there has been no South African case law which has considered the scope of such protection since 2006.[2] Coincidentally, that was also the year in which the last academic article in South Africa analysing the scope of copyright protection of computer programs was published.[3] In that, otherwise well-written, article De Villiers states that, unlike the US courts, the UK and South African courts have not

★ This article is based on part of my doctoral dissertation at Stellenbosch University. I would like to thank my supervisor, Professor Owen Dean, for his valuable guidance during my research.

† BSc LLB (UCT) LLM (London) LLD (Stell).

[1] *Northern Office Microcomputers (Pty) Ltd & others v Rosenstein* 1981 (4) SA 123 (C) at 134. This case protected computer programs as literary works. It was not until 1992, following an amendment to the Copyright Act 98 of 1978, that computer programs were protected as a distinct category of copyright work.

[2] *Haupt t/a Soft Copy v Brewers Marketing Intelligence (Pty) Ltd & others* 2006 (4) SA 458 (SCA).

[3] Roux de Villiers 'Computer programs and copyright: The South African perspective' (2006) 123 *SALJ* 315. While there have been articles, and sections in textbooks, written on the copyright protection of computer programs since De Villiers's article, the remarks therein relating to the scope (or appropriate scope) of copyright protection of computer programs have, with respect, been perfunctory. As should become clear during the course of this article, the scope of copyright protection of computer programs concerns that which is capable of being protected by copyright. It requires an examination of the elements of a computer program, and not simply a statement of the exclusive acts which are reserved for the copyright owner (see s 11B of the Copyright Act 98 of 1978).

drawn a proper distinction between functional works — such as computer programs — and other copyright works, with the result that they are more likely to protect ideas, rather than their particular expression.[4] The appropriate scope of copyright protection is central to the economic rationale of copyright law; that is, to incentivise (stimulate) the creation of copyright works, which is, arguably, the justification for copyright. If the scope of protection is too broad, the incentivising goal will be undermined as such protection will limit access by other authors to the broader themes and concepts used to create such works, which would offset the social benefit of protecting such works.

While De Villiers's statement may have been a fair comment in respect of the South African cases which have dealt with alleged copyright infringement of computer programs (principally due to the dearth of case law), it is submitted that the case of *Navitaire Inc v easyJet Airline Company & another*[5] represented a sea-change in UK copyright law relating to computer programs, and that his comment in relation to the UK position (and, arguably, by extension the South African position[6]) is, and was at the time of its publication, incorrect.[7] His article did not contain a reference to the *Navitaire* case, which, arguably, continues to be the seminal case concerning the scope of copyright protection afforded to computer programs. It would, therefore, be remiss to let De Villiers's assertion concerning the scope of protection of computer programs, as functional works, in the UK (and, by extension, also probably in South Africa), be the last word on the matter.

Accordingly, this article will examine the developments relating to the scope of copyright protection of computer programs, and explain why such protection is more limited than in the case of other, more expressive, copyright works (or, at least, not as extensive as De Villiers suggests).

II COMPARATIVE ANALYSIS

Given the dearth of South African case law on the subject, there will be particular emphasis in this article on the legal developments concerning the scope of copyright protection of computer programs in the US, and, as

[4] Ibid at 332.

[5] [2004] EWHC 1725 (Ch).

[6] The reasons for suggesting that the position in South Africa should be the same as in the UK are the historical connection, and similarities, between the South African and UK copyright laws, which makes UK case law on the scope of copyright protection of computer programs particularly persuasive and instructive.

[7] It is not clear why De Villiers's article made no reference to the *Navitaire* case supra note 5, which appears to have been published by the time of its publication. Perhaps the article had already been submitted for publication at the time the judgment was published. His statement may have been true at the time of writing (see *IBCOS Computers Ltd & another v Barclays Highland Finance Ltd & others* [1994] FSR 275 at 290). The next significant judgment recognised a distinction between functional works and other works (*Cantor Fitzgerald International v Tradition (UK) Ltd* [2000] RPC 95 at 130).

already mentioned, the UK. The US and the UK are the leading jurisdictions concerning copyright protection of computer programs, and software development. Besides the historical links between South African and UK legislation, the other reason for focusing on the developments in the UK is that the UK Copyright, Designs and Patents Act ('UK CDPA')[8] had to be harmonised with the EU Software Directive.[9] The EU Software Directive is the most recent multi-national instrument, reflecting a broad consensus, on the scope of copyright protection of computer programs. Although the US courts were the first to lead the way in considering the scope of copyright protection of computer programs, the most recent cases have been those decided in the UK, which have been significantly influenced by the EU Software Directive.[10]

Although jurisdictions like the US and the UK protect computer programs as a form of literary work, and not as a sui generis category of copyright work as in South Africa, this does not affect the persuasive authority of such case law in providing guidance on the appropriate scope of copyright protection in South Africa. South Africa's decision to protect computer programs as a sui generis category of copyright work makes it unique, as all the leading jurisdictions with computer software industries have opted for the protection of computer programs as literary works.[11] This decision was criticised at the time it was taken: Pistorius suggested that computer programs were not significantly different from other utilitarian literary works, and their functionality was, also, irrelevant.[12] It is, however, submitted that computer programs are, indeed, significantly different from other literary works, as will be illustrated below. Although jurisdictions like the US and the UK protect computer programs as literary works, computer programs are only nominally literary works; for all intents and purposes, on closer inspection, computer programs in those jurisdictions are, in reality, a distinct category of copyright work.

In fact, the very reason why determining the appropriate scope of copyright protection of computer programs has proven to be such a thorny issue may, in large part, be ascribed to the peculiar nature of computer programs. This special character of computer programs has now been

[8] Copyright, Designs and Patents Act, 1988 ('UK CDPA').

[9] Directive on the legal protection of computer programs 2009/24/EC. The original directive was adopted on 14 May 1991 (Directive on the legal protection of computer programs 91/250/EEC), and was replaced by the current version, adopted on 23 April 2009, with effect from 25 May 2009. The two versions are substantially the same, as the later version was simply a consolidation of the earlier version and its amendments.

[10] See, for example, *Navitaire Inc v easyJet Airline Company* supra note 5 para 93; *Nova Productions Ltd v Mazooma Games Ltd & Others* [2007] EWCA Civ 219 para 27 and *SAS Institute Inc v World Programming Ltd* [2013] EWCA Civ 1482 para 76.

[11] T Pistorius & C Visser 'The Copyright Amendment Act 125 of 1992 and computer programs: A preliminary overview' (1992) 4 *SA Merc LJ* 346 at 348.

[12] T Pistorius 'The copyright protection of computer programs: Literary works shunned by the proposed Bill' 1991 *De Rebus* 833 at 833–4.

recognised in US and UK case law. The early cases did not appropriately determine the scope of copyright protection of computer programs due to the 'uncritical' application of the copyright principles applicable to traditional literary works to computer programs.[13]

Contrary to Pistorius's suggestion, the courts have now accepted that the problem of determining the appropriate scope of copyright protection presented by computer programs is 'fundamentally different' from other literary works due to their functional character. Therefore it has been claimed, not without some justification, that copyright protection of computer programs was misplaced; computer programs 'are effectively a cuckoo in the copyright nest'.[14] In other words, computer programs do not fit comfortably within the copyright framework — let alone being comparable to other literary works. Determining the appropriate scope of copyright protection of computer programs, from a legal and economic perspective, requires a different calculus from other literary works — even other utilitarian literary works.[15]

Accordingly, it is suggested that no significance should be attached to the fact that South Africa protects computer programs as a sui generis category of copyright work, while the US and the UK protect them as literary works; it is not doctrinally problematic to consider US and UK case law because the case law from those jurisdictions indicates that computer programs are treated, de facto, as a separate category of copyright work. In fact, South Africa's decision to categorise computer programs as a sui generis type of copyright work is, arguably, doctrinally more acceptable as it reflects the de facto position in the US and the UK.[16]

For the sake of completeness, and contrary to De Villiers's opinion,[17] it is further submitted that the protection of computer programs as a sui generis category of copyright work is in compliance with South Africa's obligations, as a member of the World Trade Organisation, and signatory to the General Agreement on Tariffs and Trades ('GATT'), pursuant to art 10 of the Agreement on Trade-Related Aspects of Intellectual Property Rights ('TRIPS'), Annexure 1C of GATT. Article 10 of TRIPS requires that computer programs be protected as literary works as defined in the Berne

[13] See, for example, *John Richardson Computers Ltd v Flanders* (No 2) [1993] FSR 497 at 558–9 and *Cantor Fitzgerald International v Tradition (UK) Ltd* supra note 7 at 130.

[14] S E Gordon 'The very idea! Why copyright law is an inappropriate way to protect computer programs' (1998) 20 *EIPR* 10. See also P G Spivak 'Does form follow function? The idea/expression dichotomy in copyright protection of computer software' (1988) 35 *UCLA LR* 723 at 728.

[15] *Lotus Development Corp v Borland International Inc* (1995) 49 F 3d 807 at 819–20.

[16] This is not to deny the fact that it does create the potential problem of separating the authorship (or ownership) of the preparatory material from that of the computer program (see, for example, L Tong 'Copyright protection for computer programs in South Africa: Aspects of *sui generis* categorization' (2009) 12 *Journal of World Intellectual Property* 266 at 272 and 274).

[17] De Villiers op cit note 3 at 326.

Convention. The definition of 'literary and artistic works' in the Berne Convention has a very wide meaning, and includes musical works, dramatic works and cinematograph films. Although most jurisdictions protect some of these works as distinct categories, it is commonly accepted that this still complies with the requirements of the Berne Convention.[18] Therefore, in principle, protecting computer programs as a distinct category of copyright work should comply with art 10(1) as the intention is, arguably, simply that the various works included within the definition of literary work should be protected to the same extent, or in the same manner, as traditional literary works.[19] There is also support for such an interpretation based on the historical development of art 10 of TRIPS.[20]

III COMPUTER PROGRAMS

Before embarking on the legal analysis of copyright protection of computer programs, it is necessary to provide a description of what computer programs are, and how they function. An understanding of the general nature of computer programs, and the basic concepts of computer science, will assist with a proper analysis of the legal issues.[21]

A computer program is simply a set of ordered, unambiguous instructions to be performed by a computer.[22] This is also essentially the definition of 'computer program' contained in the South African[23] and US[24] copyright legislation.[25] It should be noted that 'computer program' is the protected category of copyright work, not 'computer software'. Software generally refers to more than the computer program. In essence, computer programs process data (input data) and produce output data, which data could, for

[18] For example, in the UK the three types of works are protected as distinct categories, whereas dramatic works are protected as literary works in South Africa.

[19] While it is the case that if computer programs had been protected as literary works in South Africa, computer programs may have enjoyed a longer period of protection — the life of the author plus fifty years, as opposed to a period of fifty years from when the computer program is first published or is made available to the public (s 3(2)*(b)* of the Copyright Act) — this is not economically significant. The period over which computer programs have a commercial value is much shorter than fifty years.

[20] Tong op cit note 16 at 270–1.

[21] Spivak op cit note 14 at 724.

[22] R White *How Computers Work* 7 ed (2004) 66. See also D Appleman *How Computer Programming Works* 1 ed (2000) 3–5; Pistorius & Visser op cit note 11 at 346; P I Kravetz 'Copyright protection of computer programs' (1998) 80 *J Pat & Trademark Off Society* 41 at 44; Spivak op cit note 14 at 727.

[23] Copyright Act, s 1.

[24] Section 101 of the Copyright Act 1976, Title 17 USC.

[25] Neither the UK legislation (UK CDPA), nor the EU Software Directive (Directive on the legal protection of computer programs 2009/24/EC) contain a definition of computer program.

example, consist of text or images.[26] Often, some of these input data are stored in files associated with the computer program that processes such data. The term 'computer software' thus refers to the computer program and the associated stored data.[27] The distinction is important as the data — input or output — may constitute separate copyright-protected works, but would not qualify as computer programs.[28] For example, part of the input data which a computer program may process may consist of a series of artistic works, which may, in turn, be combined with music to produce another copyright work, such as a cinematograph film. The artistic works, music, and cinematograph film are protectable as distinct copyright works.[29] For example, a program's user interface (also referred to as the 'UI' or the 'look and feel' of a computer program) is part of the output data of a computer program, and, as such, the product (or result) of the program. This distinction between a computer program's user interface and the underlying program is now generally recognised, and accepted. The European Court of Justice has confirmed that a graphical user interface does not constitute a form of expression of a computer program within the meaning of article 1(2) of the EU Software Directive, and, therefore, the copyright in the computer program does not protect its user interface; the user interface is not protectable *as* a computer program.[30]

Computers only process instructions in binary digits or 'bits': 0s and 1s. Accordingly, all instructions, or the data to be processed, must be reduced to binary form — strings of 0s and 1s.[31] However, due to the practical difficulties of specifying computer instructions in binary form — a sophisti-

[26] Appleman op cit note 22 at 21 and White op cit note 22 at 66. (See also *http://www.differencebetween.net/technology/difference-between-software-and-program/*, accessed on 21 November 2010.)

[27] Sometimes computer software is restrictively defined to correspond to the definition of 'computer program' in the South African Copyright Act (C Reynolds & P Tyman *Principles of Computer Science* 1 ed (2008) 3), while at other times it is even more extensively defined to include any related documentation and operating manuals (D M Davidson 'Protecting computer software: A comprehensive analysis' (1983) 23 *Jurimetrics J* 337 at 340–1).

[28] De Villiers op cit note 3 at 316. For completeness, it is necessary to clarify that screen displays are part of the output data of a computer program, and, as such, the product (or result) of the program; they are not computer programs. In the past, similarities in the user interfaces (pursuant to the so-called look-and-feel protection afforded to computer programs) were wrongly used as a shortcut to support a finding of copyright infringement of the underlying computer program, as a protectable non-literal element. See *Whelan Associates Inc v Jaslow Dental Laboratory Inc* (1986) 797 F 2d 1222 at 1244.

[29] Davidson op cit note 27 at 373.

[30] L J Smith 'Whether copyright protects the graphic user interface of a computer programme' (2011) 17 *CTLR* 70 at 70–1; *Bezpenostní softwarová asociace – Svaz softwarové ochrany v Ministerstvo kultury* 2010 Case C-393/09.

[31] Technically, computer programs are not converted directly into a file in binary form but go through a process of transformations, which finally gets processed as electrical impulses, corresponding to the 0s and 1s.

cated program could easily consist of hundreds of instructions, and, therefore, thousands of bits, which would be difficult for humans to work with — more convenient tools were developed for creating computer programs. Programming languages were developed to allow programmers to develop computer programs in 'high-level languages': a mixture of rudimentary English words and algebraic instructions.[32] Because of the ease with which high-level languages can be understood and used by humans, they greatly facilitate the development of computer programs; they enable programmers to more easily construct and follow the logic of a computer program, and allow for speedier and more concise creation of computer instructions.[33] For example, programmers are able to provide useful, more easily understood descriptive names for the different elements of a computer program, such as subroutines, modules, functions, procedures or variables.[34] Some of the more well-known high-level programming languages are Basic, Fortran, Cobol, C++, and Java.

The computer program written in a high-level language is referred to as the 'source code' of the program.[35] In order to be executable, the source code must be converted into 'object code' (that is, the required 0s and 1s, also called 'binary code,' 'machine code,' 'machine language,' or 'executable code').[36] This conversion of the source code to object code is performed by computer programs called compilers or interpreters.[37]

[32] J F Banzaf 'Copyright protection for computer programs' (1966) 14 *Copyright L Symp* 118 at 157–8; Kravetz op cit note 22 at 45; C Petzold *Code: The Hidden Language of Computer Hardware and Software* (2000) 352–3. The use of these programming languages is made possible as a consequence of computer programs which help to translate them into the necessary binary form. Before the development of high-level languages, programmers had developed a more basic, low-level language, assembly language, which allowed programming using simple command words and hexadecimal code to represent individual computer commands. While assembly language is still useful because it produces computer programs which are smaller and require less processing time, its use is limited. In any event, the legal analysis relating to computer programs developed in high-level languages should be equally applicable to programs developed in assembly language. See Appleman op cit note 22 at 14 and White op cit note 22 at 79.

[33] Davidson op cit note 27 at 368; Petzold op cit note 32 at 353.

[34] Davidson ibid at 378. Subroutines, modules, functions or procedures are sets of instructions in a computer program which are executable as distinct units. They are sometimes referred to as sub-programs, and could be considered as independent computer programs. A complex computer program generally consists of a master program — the kernel — which coordinates the interaction of various subprograms. Variables are essentially the data which are to be processed by a computer program. See Appleman op cit note 22 at 19 and 59; White op cit note 22 at 80.

[35] Strictly speaking, source code is the computer code produced by the computer programmer, in whichever form, a high-level language or directly in binary form (object code). However, as programming is rarely done in object code, references to the source code of a computer program are generally to computer code in a high-level language.

[36] This is a simplification as object code and machine code are not, technically, the same thing. Object code, although closely resembling machine code (comprising of

Although only the object code of a computer program is required to use it, the source code of a computer program, which generally also contains explanatory comments inserted by programmers, is more valuable to anyone seeking to establish how the program works as it is readily understandable by a suitably-trained, or skilled, computer programmer. Therefore, the source codes are usually kept confidential and the programs are distributed only in object code.[38] While the compilation of the source code into object code is a relatively simple process, the object code is not easily reversible (or 'decompiled' in computer parlance) into source code.[39] During the compilation process of converting source code to object code, programmers' explanatory comments are ignored; thus, any decompilation will not yield the helpful comments which may have accompanied the source code. The compilation process also removes the descriptive names of functions, subroutines, procedures or variables, which are replaced with symbolic representations. Furthermore, the logical order of the source code may not be apparent from the object code, which is more concerned with the order of execution of the program, rather than its design logic.[40] Thus, for anyone interested in more than using a computer program — for example, modifying the program — for all practical purposes, the source code is required, since reconstructing it requires painstaking reverse-engineering.[41]

Henceforth, the more general term 'computer code' will be used when dealing with matters applicable to both the source code and object code of a computer program, and where, from a legal perspective, no distinction is required.

bits), is not in a form which is directly executable and still needs further processing to convert it into machine code. For purposes of the legal analysis, object code and machine code can be regarded as equivalent, as is commonly the case in the literature relating to the copyright protection of computer programs.

[37] Appleman op cit note 22 at 149. It is not necessary, from a legal perspective, to elaborate on the technical distinction between the interpretation and compilation of the source code of a computer program, suffice to say that most commercial programs are compiled. Also, the interpretation or compilation of source code, technically, converts source code into an intermediate language, and not directly into object code. However, this work will simply refer to compiled programs when referring to the conversion of source code to object code. See White op cit note 22 at 86.

[38] S M McJohn 'The paradoxes of free software' (2000) 9 *George Mason LR* 25 at 27–8; C H Nadan 'Open source licensing: Virus or virtue?' (2002) 10 *Texas Intell Prop LJ* 349 at 351; Spivak op cit note 14 at 728–9.

[39] McJohn ibid at 27–8.

[40] Davidson op cit note 27 at 378. In fact, the compiled intermediate language is specifically designed to facilitate efficiency. It can be rearranged and optimised for efficiency, and in this manner, programs with different source codes can result in the same low-level machine code. As indicated above, the compilation process produces intermediate language, which is suitable for optimisation, and this is what decompilation of the object code will reconstruct — not the source code.

[41] McJohn op cit note 38 at 27.

(a) Functional works

The distinction between functional and non-functional works has been recognised by the UK courts as far back as the 19th century. This was done by either denying that the alleged copyright infringement had occurred (that is, providing limited (or 'thin') copyright protection) or denying the existence of copyright in a functional work. In *Hollinrake v Truswell* a printed sleeve chart containing words and figures, used to measure accurately the correct proportions of the inner, and outer, parts of a sleeve, was held to be a measuring tool or apparatus — 'a mechanical contrivance' — not protectable by copyright.[42] It was held that copyright protection did 'not extend to ideas, or schemes, or systems, or methods; it is confined to their expression; and if their expression is not copied the copyright is not infringed'.[43] The exceptions listed in the above quotation from the *Hollinrake* case were codified in US copyright law pursuant to s 102*(b)* of the US Copyright Act.[44]

Despite this long-standing appreciation of the difference between functional and non-functional works, the early case law concerning the scope of copyright protection of computer programs did not always bear this distinction in mind.

(b) The functional nature of computer programs

Copyright legislation, as a general rule, does not determine the scope of copyright protection. The courts have been left with the task of defining its scope. This has allowed the law to be dynamic: it can respond to the challenges brought about by rapid changes in technology and its impact on copyright doctrine. The functional nature of computer programs poses challenges to copyright law. It necessitates a careful assessment of what exactly the courts are asked to protect: the assessment of whether a substantial part of a computer program has been copied should identify — and disregard — any unprotectable elements, and give the appropriate weight to those elements which ought to be entitled to only limited copyright protection. The case law from the UK[45] and US[46] indicates that the courts did not

[42] *Hollinrake v Truswell* [1894] 3 Ch 420 426 and 428.

[43] Ibid at 427.

[44] *Lotus Development Corp v Borland International Inc* supra note 15 at 816–7; *Whelan Associates Inc v Jaslow Dental Laboratory Inc* supra note 28 at 1234. Section 102*(b)* provides as follows: 'In no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work.'

[45] See, for example, *IBCOS Computers Ltd & another v Barclays Highland Finance Ltd & others* supra note 7 and the criticism thereof by K Tumbraegel & R de Villiers 'Copyright protection for the non-literal elements of a computer program' (2004) 10 *CTLR* 34.

[46] See, for example, *Whelan Associates Inc v Jaslow Dental Laboratory Inc* supra note 28 and the criticism thereof by Spivak op cit note 14; and L Green 'Copyright protection and computer programs: Identifying creative expression in a computer program's nonliteral elements' 1992 *Fordham Ent Media & Intell Prop LJ* 3 at 89.

initially appreciate fully the nature of computer programs, and too readily sought analogies with traditional literary works.

Issues concerning the scope of copyright protection have invariably arisen in cases involving the alleged infringement (particularly, the unauthorised copying) of a computer program. Infringement by way of copying will take place if there is sufficient objective similarity between the alleged infringing computer program and the original program, and there is a causal connection between the original program and the creation of the alleged infringing program.[47] It is important to bear in mind that copyright infringement does not require that an entire copyright work be copied; it is sufficient if *any* substantial part of the work is copied.[48]

Copyright-infringement cases involving computer programs can be divided into two broad categories: cases involving literal (or textual) copying, and those involving non-literal (or non-textual or non-code) copying. Literal copying involves the verbatim copying of a program's computer code, or any substantial part thereof. Included within this category would be those cases were computer code has been copied and only colourable alterations have been made thereto to disguise the copying, or where a computer program has simply been translated into a different programming language. The *Haupt* case from 2006 referred to in the introduction was principally concerned with literal copying.

Non-literal copying is not a term of art, and there have been various definitions of what constitute the non-literal components of a computer program.[49] For example, in the *Navitaire* case, the court adopted a very narrow definition of non-literal copying: it comprised those instances of alleged copyright infringement where there had been no access to the computer code.[50] However, for purposes of this article, non-literal copying will be defined as any alleged copyright infringement which is not literal copying. For convenience, this means any alleged copyright infringement that does not involve the literal copying of a program's computer code (or any substantial part thereof), does not involve only colourable alterations to disguise such verbatim copying, or which is a mere translation of a program into another programming language.

[47] *Galago Publishers (Pty) Ltd & another v Erasmus* 1989 (1) SA 276 (A) at 280.

[48] South African Copyright Act, s 1(2A). Similarly, the doing of an unauthorised restricted act in respect of any substantial part of copyright work will also constitute copyright infringement (s16(3)of the UK CDPA). The US equivalent is 'improper appropriation', which is the requirement that a material portion of the copyright work must have been copied. Such a determination will disregard the copying of facts, general ideas, and unoriginal expression. See M LaFrance *Copyright Law: In a Nutshell* 2 ed (2011) 284–5.

[49] B Bordoloi, P Ilami, P P Mykytyn et al 'Copyrighting computer software: The "look and feel" controversy and beyond' (1996) 30 *Information & Management* 211 at 214.

[50] *Navitaire Inc v easyJet Airline Company* supra note 5 para 113.

The reason for drawing a distinction between literal and non-literal copying is that, as a general rule, the cases concerning literal copying are not very illuminating about the scope of copyright protection of computer programs. This was, for example, the situation in the *Haupt* case. Cases concerning non-literal copying, and specifically the determination of whether any substantial part of a computer program may have been copied, have required the courts to grapple with the nature of computer programs, in order to ensure that their protection does not extend beyond incentivising their creation. For example, an area in which the functional nature of computer programs has caused problems is when trying to distinguish ideas from their expression.[51] The idea-expression dichotomy is a fundamental doctrine of copyright law. Copyright does not protect ideas; only particular expressions of ideas are protected.[52]

Of the three jurisdictions, the US courts have, traditionally, not shied away from the idea-expression dichotomy, possibly because US courts have been more acutely aware of the economic effects of copyright law, and its social purpose. For example, in the *Whelan* case — the first important US case concerning the appropriate level of copyright protection for non-literal elements of a computer program — the purpose, and importance, of the idea-expression doctrine in copyright law was made clear.[53] The idea-expression dichotomy in copyright law has its basis in the purpose of the copyright law, which is to strike an efficient balance between incentivising the creation of copyright works, and the dissemination of information, to promote learning, culture and development.[54]

The determination of what will be regarded as protectable expression 'may depend on the type of idea being expressed, the medium of expression, and even the value of the idea to society'.[55] The separation of protectable expressions from unprotectable ideas involves the identification of a series of abstractions: at one extreme the abstraction may be of such a high level that it merely describes the general nature of the copyright work, and, at the other extreme, there is simply the specific expression. A court has to determine the specific protectable abstraction between these two extremes.[56] The difficulties presented by the application of the idea-expression doctrine to traditional

[51] See, for example, *Whelan Associates Inc v Jaslow Dental Laboratory Inc* supra note 28 at 1235–6; *Computer Associates International Inc v Altai Inc* (1992) 982 F 2d 693 at 702–4; *John Richardson Computers Ltd v Flanders (No 2)* supra note 13 at 523; *IBCOS Computers Ltd v Barclays Highland Finance Ltd* supra note 7 at 290–1; *Navitaire Inc v easyJet Airline Company* supra note 5 para 94.

[52] The idea-expression distinction is also recognised in TRIPS (art 9(2)) and in the EU Software Directive (art 1(2), and recitals 13 and 15).

[53] *Whelan Associates Inc v Jaslow Dental Laboratory Inc* supra note 28 at 1235.

[54] Ibid.

[55] Banzaf op cit note 32 at 148–9.

[56] T Pistorius 'The copyright protection of computer programs in the United States of America: The second generation questions (part 2)' (1992) 25 *De Jure* 166 at 170.

copyright works, such as literary or artistic works, are compounded when it has to be applied to computer programs. Computer programs are both functional and expressive, and 'ideas are closely intermingled with the expression thereof'.[57] Much of the case law concerning alleged non-literal infringement of computer programs has centred on the issue of trying to distinguish unprotectable ideas (or elements) from protectable expression.

Although computer code may be readable by humans, and, to that extent, resembles literary works, this belies its true purpose; the primary function of computer code is to be executed by a computer in order to realise the purpose of the particular computer program. This qualitative, functional difference distinguishes it from other forms of intellectual property, such as creative works or patents, which do not have such a direct, and purposive (or literal, rather than literary) character. For example, a musical work does not, by itself, create music, and a patent does not create the patented item.[58] The computer code of a computer program is not just symbolic — like an architect's drawing is symbolic of the building to be constructed, or a cake recipe in relation to the cake described — it is also mechanical in the sense that it operates in a direct manner. A computer program does not simply provide instructions for, or reveal, how a computer will work; it actually makes the computer work in the specified manner.[59] Although the source code still has to be decompiled into object code, this involves a routine process (requiring no additional, protectable human effort). Given the fact that source code has to conform to strict syntactical and semantic criteria in order to be executable, it is, at least from a legal perspective (and, practically speaking, from a technical perspective), functional in nature. The quality, and accuracy, of the source code will determine whether the program will function correctly, or at all.

Computer instructions must conform to the strict syntactical and semantic criteria of the chosen programming language if they are to be executed. In this sense, every portion of the computer code of a computer program is critical to its operation. This, however, does not mean that every part can be considered a substantial part of a computer program for purposes of copyright law.[60] Similarly, substantiality does not depend on the extent to which a particular portion of the computer code is used.[61] Moreover, although there may be differences between programming languages, there are still significant similarities, and they all rely on the arrangement of Boolean operations.[62] Whether the particular portion taken constitutes a substantial portion

[57] Tumbraegel & De Villiers op cit note 45 at 35.
[58] Davidson op cit note 27 at 344.
[59] Ibid at 345.
[60] *Cantor Fitzgerald International v Tradition (UK) Ltd* supra note 7 at 130–1.
[61] Ibid at 135.
[62] Appleman op cit note 22 at 37. Boolean logic involves a determination of the truth values of a sequence of operations ('and' (conjunction), 'or' (disjunction), and 'not' (negation)) applied to statements which can have one of two values, 'true' (1) or 'false' (0).

of the copyrighted program should depend on the skill and labour involved in its creation.[63]

IV THE SCOPE OF COPYRIGHT PROTECTION

Copyright law has dealt with issues of non-literal copying in other contexts, such as fictional works. If copyright protection was confined to literal copying of a fictional work, copyright protection would provide the author very little protection against a person who skillfully paraphrases his work. It would, for example, allow a plagiarist to avoid liability by making immaterial variations to the copied work. Copyright protection, therefore, necessarily has to protect more than just the literal text (or translation) of a copyright work.[64] Accordingly, courts have held that the non-literal copying of the plot, or plot devices, of a play or book can constitute copyright infringement.[65] This approach has been recognised by the courts: copying that paraphrases, or loosely paraphrases, the computer code of a computer program, rather than taking the verbatim expression, can constitute copyright infringement.[66] However, given the functional nature of computer programs, the scope of protection should not be so broad that it protects the underlying ideas, or inhibits future creative endeavours by others.[67]

Given the fact that a computer program can for some purposes, such as part of the development process, be considered in layers of elaboration and gradual refinement at various levels of abstraction — moving from a general idea to a specific application — these levels of abstraction have been used to determine the scope of copyright protection of computer programs, to distinguish unprotectable ideas from protected expression, on the basis of the idea-expression dichotomy in cases of non-literal copying.[68] A typical example of this form of analysis by analogy was the search for the structure (or sequence, or organisation) of a computer program.[69] The purpose of a utilitarian work, such as a computer program, was considered to be the work's idea, and everything that was not necessary for that purpose was

[63] *Cantor Fitzgerald International v Tradition (UK) Ltd* supra note 7 at 135. See also A Murray *Information Technology Law: The Law and Society* (2010) 196. Similarly, our courts have accepted that a computer program is original and, therefore, eligible for copyright protection if its creation requires skill, judgment, or labour (*Haupt t/a Soft Copy v Brewers Marketing Intelligence (Pty) Ltd* supra note 2 at 473).

[64] Banzaf op cit note 32 at 148–9; M A Hamilton 'Computer science concepts in copyright cases: The path to a coherent law' (1997) 10 *Harvard J on Law & Tech* 239 at 244.

[65] *Nichols v Universal Pictures Corp* (1930) 45 F 2d 119; *Sheldon v Metro-Goldwyn Pictures Corp* (1936) 81 F 2d 49; *Twentieth Century-Fox Film Corp v MCA Inc* (1983) 715 F 2d 1327.

[66] *Lotus Development Corp v Borland International Inc* supra note 15 at 814.

[67] Gordon op cit note 14 at 12; Pistorius op cit note 56 at 169.

[68] Davidson op cit note 27 at 379; Spivak op cit note 14 at 729–31; White op cit note 22 at 76.

[69] *Whelan Associates Inc v Jaslow Dental Laboratory Inc* supra note 28 at 1237.

considered to be part of the protectable expression of that idea.[70] Thus, provided that there were other computer program structures which could achieve a similar purpose, the particular detailed structure of the program was not essential to its function, and constituted protectable expression of that idea.[71]

However, this analogy with literary works such as fiction or poetry, although used by the courts in earlier decisions concerning computer programs, should, if at all, be used with extreme caution in the case of computer programs. Non-functional copyright works, like fictional literary works, are purer expressions of an author's creative mind: other than the fact that a literary work may make use of general plot lines, themes, literary techniques, or stock characters, authors are not constrained by functional considerations. In the case of fiction, or poetry, an author has greater room for individualised expression; the manner in which something is expressed in a work of fiction, or poem, says a great deal more about what is expressed (and its author), than in the case of a computer program. Hettinger sums up this situation in the case of fiction, or poetry, as follows: 'In these mediums, more so than in others, *how* something is said is very much part of *what* is said (and vice versa).'[72]

In contrast to fictional literary works, computer programs with a similar function (or addressing a particular problem) will — because of their functional nature — necessarily exhibit a greater degree of similarity. The structure of a computer program leaves much less choice for expression than creative works in human language, because a program must conform to strict syntactical and semantic criteria. The more restricted one's choices are, the less likely it is that what is created will result in an intellectual creation protectable by copyright.[73] Thus, due to the fact that computer programs will invariably have to conform to technical requirements, constrained by efficiency considerations (such as the need to manage data flow), and employ standard techniques and common expressions, a greater degree of similarity may be required before a finding of substantial similarity can be sustained.[74] Accordingly, copyright protection of computer programs may be required to be more limited.

As a result of the failure to appreciate this difference, inappropriate use of copyright concepts well-known in the context of literary works such as 'structure' and 'organisation' were applied to computer programs.[75] Use of a concept like the structure of a computer program necessarily requires a court to consider the program at a particular level of abstraction. If copyright

[70] Ibid at 1236.

[71] Ibid at 1242–3.

[72] E C Hettinger 'Justifying intellectual property' (1989) 18 *Philosophy and Public Affairs* 31 at 32.

[73] *SAS Institute Inc v World Programming Ltd* supra note 10 paras 31–3.

[74] Banzaf op cit note 32 at 154–5; Green op cit note 46 at 89; Murray op cit note 63 at 199; Pistorius op cit note 56 at 169; Spivak op cit note 14 at 755.

[75] Hamilton op cit note 64 at 240.

protection is extended to the structure of expressions at the wrong level of abstraction (if it is at all appropriate to use such a concept), it could result in broader protection than would be appropriate.[76] By defining a program's idea as its purpose, and everything that was not necessary to that purpose as part of the protectable expression, the court in the *Whelan* case favoured an expansive notion of what constitutes protectable expression.[77] This already overly-broad protection afforded to computer programs was expanded, culminating in *Lotus Development Corp v Paperback Software Intl*,[78] which provided protection to a computer program's user interface, or 'look and feel,' as one of the protected elements of a computer program.[79]

Despite the US courts initially granting overly-broad protection to computer programs, subsequent case law involving non-literal copyright infringement of computer programs significantly narrowed the protection afforded to computer programs, as exemplified by the decision in a different *Lotus Development Corp* case.[80] This change was attributable to the courts' greater appreciation of the functional nature of computer programs, and ceasing to draw inappropriate analogies between a computer program and non-functional works such as novels, and the law applicable to protecting the latter type of work. In order to deal with the difficulty in determining whether a substantial part of a computer program had been copied, alternative tests were developed to determine whether infringement had taken place.

The most well-known of these alternative tests to determine if two computer programs are substantially similar is the three-step procedure, formulated by the court in the *Computer Associates* case,[81] known as the AFC test: an abbreviation of the three sequential steps — abstraction, filtration, and comparison.[82] It requires a court first to break down the copyrighted program into its constituent structural parts. Each of these parts must then be analysed to filter out the unprotectable elements (associated ideas, those ideas dictated by efficiency or external factors, and public domain elements) to determine the protectable kernel (the 'golden nugget') of the program.[83]

[76] Ibid at 273.

[77] R Arnold 'Infringement of copyright in computer software by non-textual copying: First decision at trial by an English court' (1993) 15 *EIPR* 250 at 252; Green op cit note 46 at 5; Spivak op cit note 14 at 748.

[78] *Lotus Development Corp v Paperback Software Intl* (1990) 740 F Supp 37. The protectability of the user interface of the Lotus 1-2-3 spreadsheet program was also the subject of the case which reversed this trend, *Lotus Development Corp v Borland International Inc* supra note 15.

[79] T Hill 'Fragmenting the copyleft movement: The public will not prevail' 1999 *Utah LR* 797 at 805.

[80] *Lotus Development Corp v Borland International Inc* supra note 15.

[81] *Computer Associates International Inc v Altai Inc* supra note 51 at 693.

[82] Ibid at 706.

[83] Ibid.

This protected expression must then be compared to the structure of the allegedly infringing work to determine if they are substantially similar.[84]

There has also been a corresponding narrowing of the scope of protection of the user interface of computer programs. Although the user interface is not a computer program, as already mentioned above, to the extent that an element of the user interface is of a functional nature, it too has limited protection. For example, in *Lotus Development Corp v Borland International Inc* a menu command hierarchy was held to amount to a method of operation — that is, the means by which an end user operates the computer program — and which was thus unprotectable.[85] Protecting a method of operation would allow the copyright holder of a program to benefit from the investment made by users in learning how to use it, rather than its own investment. The users' investment could lead to the copyright holder having a monopoly, not because it produced a better program or because it represented better value.[86] In the *Lotus* case the court stated that the functional nature of computer programs does not prevent them from being copyrightable, but it does change how they should be assessed in terms of copyright doctrine.[87]

There has, broadly speaking, been a similar development concerning the scope of copyright protection of computer programs in the UK. Despite the UK courts in the first significant English case concerning non-literal infringement, *John Richardson Computers Ltd v Flanders (No 2)*, stating that the approach set out in *Computer Associates* was generally consistent with English law,[88] the English courts subsequently rejected that approach, leading to a widening of the scope of protection.[89] This unfortunate tendency was, thankfully, reversed, with a greater appreciation for the functional nature of computer programs in the UK.[90] The *Navitaire* decision finally signalled a full appreciation for the functional nature of computer programs, and resulted in the recognition of the thin protection afforded to computer programs, resulting in a position similar to that which exists in the US.

V THE *NAVITAIRE* DECISION

The case concerned, amongst other things, the alleged infringement of the claimant's (Navitaire) software for a 'ticketless' airline booking system called OpenRes.[91] It was claimed that a similar system developed for easyJet Airline Company ('easyJet'), called eRes, infringed the copyright in the OpenRes

[84] Ibid.
[85] *Lotus Development Corp v Borland International Inc* supra note 15 at 815.
[86] Ibid at 821.
[87] Ibid at 819.
[88] *John Richardson Computers Ltd v Flanders (No 2)* supra note 13 at 526–7.
[89] See *IBCOS Computers Ltd v Barclays Highland Finance Ltd* supra note 7.
[90] *Cantor Fitzgerald International v Tradition (UK) Ltd* supra note 7.
[91] *Navitaire Inc v easyJet Airline Company* supra note 5 para 1.

system.[92] Navitaire did not allege that easyJet, or BulletProof (the company that developed eRes for easyJet), had access to the source code of its OpenRes system.[93] It was alleged that the overall similarity in the user functionality of the OpenRes and eRes systems amounted to non-textual copying of the OpenRes system;[94] that the 'business logic' of the OpenRes system had been appropriated.[95]

The court held that despite the similarities in the appearance of the two systems to users, the underlying processes by which the two systems operated were different.[96] Because a computer (along with its computer program) is a deterministic machine, it was possible to identify the specific responses to various inputs, and to replicate such behaviour by writing another appropriate program.[97] In essence, Navitaire contended that the writing of such a second program infringed the copyright in the source code of the first program.[98] However, it was clear that easyJet and BulletProof had no access to the source code of the OpenRes system, and that there were differences in the programming languages, computer code and architecture between the OpenRes and eRes systems.[99] The court also did not consider the eRes computer code to be a translation or adaptation of the OpenRes code.[100]

If Navitaire was going to succeed in its claim, it needed to show that, at some level of abstraction, the defendants copied something (other than the command set and the screen displays) that was not inherent in the nature of the business function to be performed by the software.[101] The analogy with a plot of a novel was a poor one because in this case the alleged copyist did not have access to the copyright work; that is, the computer code.[102] In any event, the notion that the overall functioning of a computer was analogous to, and protectable in a manner similar to, the plot of a novel, was incorrect. Unlike the plot of a novel, the user interface was not part of the work itself as the same result could be achieved by different computer programs, involving no copying. [103] Computer programs do not have themes, events, plots or narrative flows; they are process driven in order to achieve a particular outcome. There was no discernible business logic which could be identified from the computer code of a computer program.[104]

[92] Ibid.
[93] Ibid para 2.
[94] Ibid.
[95] Ibid paras 73 and 108.
[96] Ibid para 110.
[97] Ibid para 112.
[98] Ibid.
[99] Ibid para 113.
[100] Ibid.
[101] Ibid para 118.
[102] Ibid para 125.
[103] Ibid para 94.
[104] Ibid para 125.

The court — rather unnecessarily — still nominally considered it appropriate to support the view expressed in the *IBCOS* case, that US copyright law was different from the UK law; it was said that US law drew a distinction between ideas and expression, and did not provide protection to functional works.[105] However, the court was forced to acknowledge that even if this was historically correct, the position had now changed: copyright protection should not be extended to the functional aspects of a computer program, such as the user interface.[106] Although this necessarily required drawing a line between idea and expression in a particular place, which may be regarded as providing too little protection for a particular expression, this was in accordance with the EU Software Directive.[107]

Relying on Lord Hoffmann's speech in *Designers' Guild*,[108] the court stated that the idea-expression distinction applies in two types of situations. First, a copyright work which describes a system, concept or invention was not infringed by the use of that which was described.[109] Secondly, a particular idea incorporated in a work (or aspects thereof) may not be protected because it may not be original, or so commonplace that it does not constitute a substantial part of the work. This assessment necessarily depended on a particular level of abstraction, and an assessment of the skill and labour involved in that aspect of the work.[110] The court held that the OpenRes system, as a whole, lacked substantiality and its overall functioning was not the result of relevant skill and effort: its inputs and outputs were in a form expected from a computer program carrying out that particular business function. Thus, the claim for non-literal copying failed.[111]

This position was considered to be consistent with the policy of the EU Software Directive, that is, to 'exclude both computer languages and the underlying ideas of the interfaces from protection'.[112] If protection were extended to the 'business logic' or overall function of a computer program, the provisions of the Directive could be circumvented. In any event, it would not have been appropriate to extend copyright protection to 'business logic'.[113] The EU Software Directive excludes the protection of ideas and principles which underlie any element of a program like its logic, algorithms, programming languages or interfaces.[114] The idea-expression dichotomy

[105] Ibid para 91.
[106] Ibid para 94.
[107] Ibid.
[108] *Designer's Guild Ltd v Russell Williams Textiles Ltd* [2000] 1 WLR 2416.
[109] *Navitaire Inc v easyJet Airline Company* supra note 5 para 128.
[110] Ibid.
[111] Ibid para 129.
[112] Ibid para 130.
[113] Ibid.
[114] Ibid para 87.

also means that functional aspects will not be protected where these have been independently replicated.[115]

In other words, a particular non-literal element will be protectable if at a particular level of abstraction it amounted to something original, which required skill and labour, other than the 'business logic' or overall function of the computer program.[116] The most recent decision of the European Court of Justice, the *SAS Institute* case,[117] confirmed the approach in the *Navitaire* case that it is permissible to emulate the functionality of a computer program, including compatibility with its data file formats, by simply observing its operation.

The *SAS Institute* case involved, inter alia, alleged copyright infringement of SAS Institute's SAS System, which was a statistical analytical software program which enabled users to write and run application programs ('Scripts'), in the SAS computer language, to manipulate data.[118] World Programming Limited produced software, the World Programming System ('WPS'), designed to emulate the SAS System, by producing the same outputs, based on the same inputs, as the SAS System. This enabled users to run the Scripts developed for use with the SAS System on the WPS.[119]

The ECJ held that the functionality of a computer program, the programming language and the format of data files used in a computer program in order to exploit certain of its functions do not constitute protectable expressions of a program for the purposes of art 1(2) of the Software Directive.[120] Accordingly, the reproduction of the functionality of a program by using the same programming language, and the same format of data files, by means of observing, studying and testing the program, does not constitute copyright infringement.[121]

While the keywords, syntax, commands and combinations of commands, options, defaults and iterations, consisting of words, figures or mathematical concepts, in isolation, were not protected by copyright, their choice, sequence and combination could amount to the protectable expression of the author of the computer program.[122] Accordingly, it was for the relevant national court — the English court — to determine if their choice, sequence and combination amounted to protectable expression for purposes of copyright.[123]

The English Court of Appeal held that there will not be a protectable, original intellectual creation if the expression was dictated by technical

[115] S Stokes 'The development of UK software copyright law: From John Richardson Computers to Navitaire' (2005) 11 *CTLR* 129 at 133.
[116] *Navitaire Inc v easyJet Airline Company* supra note 5 paras 128–130.
[117] *SAS Institute Inc v World Programming Ltd* supra note 10.
[118] Ibid para 23.
[119] Ibid para 24.
[120] Ibid paras 39 and 46.
[121] Ibid paras 44 and 45.
[122] Ibid paras 66 and 67.
[123] Ibid paras 69 and 70.

function.[124] This was significant, because the court accepted that the design of the SAS System, which allowed for numerous analytical options, was the result of considerable intellectual effort based on statistical analysis.[125] The functionality of a computer program (in the sense of what it does and how it responds to particular inputs) falls on the ideas side of the idea-expression divide.[126] The functionality of a computer program comprises those actions, or purposes, which a user expects it to perform.[127] Functionality amounted to an idea, and, if it were allowed to be protected as such by copyright, it would make it possible for someone to monopolise ideas.[128] This was also the case where several functionalities were combined.[129] Thus, the effort in determining the functionality of a computer program, or a specific combination of functionalities, did not give rise to copyright protection.[130] In so far as computer programs were concerned, it was not just procedures, methods of operation and mathematical concepts which were considered as falling within the scope of ideas rather than expression; so too did keywords, syntax, commands and combinations of commands, options, defaults, iterations (which all consist of words, figures or mathematical concepts), and selections of statistical operation.[131] Even if these elements involved choices relating to their selection or combination, such effort did not result in the creation of protectable expressions, which is a more restricted concept than intellectual creation.[132]

However, the particular manner in which formulae and algorithms are combined (rather than the formulae and algorithms themselves) may amount to a protectable expression if it involved the author's own intellectual creation.[133] But in this case there could be no infringement on this basis as WPS did not copy the program directly, as it had no access to the source code or the object code.[134]

In the UK, like the US, there has also been a corresponding narrowing of the scope of protection of the user interface of computer programs, as evidenced in the subsequent *Nova Productions* case.[135] In the *Nova Productions* case the court stated that the EU Software Directive makes it clear that the ideas incorporated in computer programs (which include their preparatory design material) are not protected.[136] What is not permitted is for another

[124] Ibid para 33.
[125] Ibid paras 13 and 14.
[126] Ibid para 74.
[127] Ibid paras 40 and 41.
[128] Ibid paras 40, 41 and 43.
[129] Ibid paras 44 and 45.
[130] Ibid paras 45, 46 and 47.
[131] Ibid paras 59.
[132] Ibid paras 59 and 61.
[133] Ibid para 42.
[134] Ibid para 13.
[135] *Nova Productions Ltd v Mazooma Games Ltd* supra note 10.
[136] Ibid para 50.

program to copy the specific expression of those ideas or functions, which have been reduced to a material form, as a literary work. Others are free to use the ideas or functions.[137] The court held that while this position may be considered as providing no effective protection for game developers, the fact is that most, if not all, copyright works are influenced and derived from other works, and it is important that copyright protection encourages competition by not stifling the creation of works which are actually very different.[138] If general ideas are protected, copyright would be an instrument of oppression rather than act as an incentive for creation, which is its purpose.[139]

VI CONCLUSION

As the court stated in the *Navitaire* case, the scope of copyright protection of computer programs is a matter that has 'vexed many judges'[140] and has not been definitively settled, although greater clarity has been provided. Despite the different nature of computer programs, the courts, at first, incorrectly analogised computer programs with copyright works with which they were familiar, such as novels, resulting in the scope of protection being too broad. The *Computer Associates* case's narrow characterisation of the scope of copyright protection of computer programs, and the *Lotus* case's examination of the protectability of elements of the user interface, is consistent with the current position in the UK. Not only was the comparatively thin copyright protection for computer programs, following the *Navitaire* decision, considered to strike an appropriate balance between providing the necessary incentives and a sufficiently large public domain, it has been described as the 'gold standard' test in the UK for cases of non-literal copyright infringement of computer programs.[141]

The fact is that the type of creativity involved in creating software is more akin to that associated with an engineering project than artistic creativity. Progress in such a technical field is incremental, and excessive protection prevents such an incremental, cumulative process from being realised. The likelihood that good substitutable products are likely to emerge depends, to some degree, on the scope of protection enjoyed by copyright owners. This affects the extent of a copyright owner's possible market power because such market power would be proportional to the scope of copyright protection: the narrower the scope of copyright protection, the more other works can copy a copyright work.[142] By allowing free use of ideas, general concepts and other unprotectable elements embodied in copyright works, competition is encouraged by copyright as it 'foster[s] a built-in process of 'reverse-

[137] Ibid para 51.
[138] Ibid para 54.
[139] Ibid para 55.
[140] *Navitaire Inc v easyJet Airline Company* supra note 5 para 93.
[141] Murray op cit note 63 at 209; Stokes op cit note 115 at 133.
[142] G S Lunney 'Re-examining copyright's incentive-access paradigm' (1996) 49 *Vanderbilt LR* 483 at 518–9.

engineering' that enables many independently created and copyrightable works to cluster around common themes or ideas'.[143] These exceptions are, thus, useful for potential competitors when considering whether they are permitted to develop substitute, competitive computer programs. Competitors would want their product to be as close as possible to a perfect substitute to a successful product, if not superior. Although the current level of protection essentially only prevents mechanical, or slavish, copying of computer programs, as non-literal infringement of a computer program has effectively been rejected by the courts; this limited right to prevent piracy or slavish literal copying, combined with the first-mover advantage, is sufficient to ensure that a third party does not gain an unfair advantage over the author of the original program.

Providing over-broad copyright protection to literary works, other than computer programs, only results in a small additional social cost. However, computer programs are fundamentally different from other literary works as they are instrumental in nature in that they are intended to bring about a particular result. In this respect they resemble objects of mechanical utility, which are generally protected by patent law. Although the functional nature of computer programs did not prevent computer programs from being copyrightable, it did change how they should be assessed in terms of copyright doctrine.[144]

[143] J H Reichman 'Charting the collapse of the patent-copyright dichotomy: Premises for a restructured international intellectual property system' (1995) 13 *Cardozo Arts & Entertainment LJ* 475 at 494.

[144] *Lotus Development Corp v Borland International Inc* supra note 15 at 819.